Fortran 90 at NAS: Perceptions and Plans
RND-93-001

I.E. Stockdale
Computer Sciences Corporation
NASA Ames Research Center
Moffett Field, CA 94035-1000, USA
Numerical Aerodynamic Simulation Systems Division

Abstract

The new Fortran standard, Fortran 90, has been adopted by ISO and ANSI. Fortran 90 contains all of FORTRAN 77, while adding many new features. These features include array constructs, new control constructs, dynamic memory allocation, user-defined data types, and new means for structuring sub-programs.

Computer hardware and software vendors, along with NAS users, have been surveyed regarding their plans for implementation and use of Fortran 90. Based on this information, a strategy is proposed for the adoption of Fortran 90 at NAS.

1. Introduction

Fortran has long been the *de facto* standard programming language in the scientific and numerical analysis communities. The American National Standards Institute (ANSI) and International Organization for Standards (ISO) have recently approved a new version of the language, known as Fortran 90 [*1*]. Fortran 90 replaces FORTRAN 77 [2] as ISO standard Fortran, while ANSI will regards both language versions as standards. The new standard changes the capitalization of Fortran from all upper-case to the normal capitalization for proper nouns.

Fortran 90 is a larger language than FORTRAN 77. In addition to retaining all of FORTRAN 77, it incorporates language features, such as data structures and pointers, which have been available in other programming languages and in individual vendors' implementations of Fortran. It also introduces array syntax, which is expected to be widely used in scientific computation. Again, this feature has been available in some vendors' Fortran compiling systems.

The High Performance Fortran (HPF) initiative, which is one of several efforts which seek to standardize parallel extensions to Fortran, will be a superset of the complete Fortran 90 standard. In order to expedite the adoption of HPF, HPF participants are defining a subset of Fortran 90 which must be supplied in vendors' early HPF releases.

When this definition is completed, it is expected to provide a portion of the new language which is portable between all vendors supporting HPF. The HPF subset is regarded as an *interim* measure; final HPF compilers will need to supply all of Fortran 90.

In this report, the major new features of Fortran 90 are summarized. Then follows a report on a survey of interest in Fortran 90. Both vendors and users were queried for this survey. Finally, a proposed official position on the adoption of Fortran 90 for use in computational science at NAS is presented.

2. Fortran 90 Extensions

The most important additions to Fortran in the new standard are described below. This is not an exhaustive discussion; in some instances, the list of features merely exemplifies the extensions present in Fortran 90. Complete documentation may be found in the standard [2], There are a number of books which elaborate on the standard (*e.g.* [3]).

2.1 Control constructs

Several new control constructs have been added in Fortran 90. The *CASE* construct allows the user to execute different sections of code based on the value of an integer, character, or logical variable. This statement is similar to the C *switch* statement. It allows clearer code than FORTRAN 77 *IF* blocks when a single expression is used to make selections.

The *DO WHILE* statement is used to repeatedly execute a block of code until some condition in the block, such as a function return code, changes. The same effect may also be achieved by using the *EXIT* statement, which transfers control out of the loop, with a test on the condition inside a loop. The *CYCLE* statement is used inside a loop to transfer control to the end of the loop. The *ENDDO* statement has been added as an additional way to close a *DO* loop; this eliminates the need for a statement label at the end of a loop.

2.2 Array constructs

Fortran 90 arrays have been greatly enhanced over their FORTRAN 77 predecessors. Fortran 90 arrays may be operands of numeric intrinsic operators (**, /, *, -, +), and assignment operators (=). They may also be passed to and returned from intrinsic functions, *e.g.* A = SQRT(B) where A and B are conformable arrays. Two arrays are conformable if they have the same shape; that is, the same number of dimensions with identical extent. All operations are performed element-by-element. In the example above, A(1,1,1) = SQRT(B(1,1,1)), *etc.* Zero-sized arrays have been added to Fortran 90.

The new standard also calls for *array sections*. These are portions of arrays which may be operated on in the same fashion as described for *whole arrays* above. Array sections may be referenced using *subscript triplets* or *vector subscripts*. The former is a set of three integers comprising the lower index, upper index and stride which define the array section. For example, A(1:5:2) references the array elements A(1), A(3), and A(5). A vector subscript is an integer array containing a list of indices into an array. If INDEX is a 3-element integer array containing 1, 3, and 5, then A(INDEX) references A(1), A(3), and A(5).

New intrinsic functions have been added for array manipulations. The DOT_PRODUCT and MATMULT functions have been added for vector and matrix operations. There are several matrix reduction operations, including SUM, MAXVAL, and MAXLOC. Array shapes may be changed (RESHAPE) and their contents re-arranged (TRANSPOSE, CSHIFT, *etc.*) with new intrinsic functions.

The *WHERE* construct allows the user to take actions based on the values of different array elements without explicitly referencing the elements. For example, the square root of all array elements with positive values could be coded in a single line. A *BLOCK WHERE* analogous to, but more limited than, the FORTRAN 77 *BLOCK IF*, is also available.

One-dimensional arrays may be initialized using *array constructors*, essentially adding the idea of array constants to the language. Multi-dimensional arrays are initialized by combining array constructors with the RESHAPE intrinsic function.

FORTRAN 77 allowed *assumed size* arrays. Fortran 90 implementations add *assumed shape* arrays, where both the size and the shape of an array are the same as in the calling routine. Fortran 90 also adds *automatic arrays*, which may be defined in a sub-program, and have a pre-defined shape and variable bounds.

2.3 Data structures

Using Fortran 90, the user can define his or her own data types as in other high level languages (*e.g.* C, Pascal), and in some vendors' FORTRAN implementations (*e.g.* VAX Fortran). The *TYPE* statement is used to define and declare such data structures, which may consist of one or more variables. These variables may be either scalars or arrays, and may be of any type (including other derived types). Derived data types may be initialized with constructors in the same manner as arrays.

The following code fragment defines a three-dimensional vector type, declares an variable of this type, initializes the variable, and then references the first element of the variable:

```
TYPE THREE_VECTOR
        REAL X_COORD
        REAL Y_COORD
        REAL Z_COORD
END TYPE THREE VECTOR

TYPE (THREE_VECTOR) THIS_VECTOR

THIS_VECTOR = THREE_VECTOR( 3.0, 5.5, -2.0 )

X = THIS_VECTOR%X_COORD
```

Derived types may be used for such purposes as building linked lists, or grouping together data which is relevant for a particular subroutine or section of code.

2.4 Program structures

FORTRAN 77 allowed the user to structure a program using *FUNCTION* and *SUBROUTINE* program units. Code which was needed in different program units, such as *COMMON* declarations, had to be explicitly duplicated in each FUNCTION or SUBROUTINE which used it. Fortran 90 eliminates the need for this duplication by standardizing the INCLUDE file mechanism, which was already present in most vendors' implementations. INCLUDE files contain definitions of COMMON blocks, types, *etc.* which are then pulled into individual program units with the INCLUDE command.

Dummy arguments to subroutines and functions may now be declared with *OPTIONAL* and/or *INTENT* attributes. The intent of the former is clear, while the latter specifies that arguments are input-only, output-only, or both. Unlike in FORTRAN 77, functions may be called recursively in Fortran 90.

Programmers desiring more structure than is provided by the combination of subroutines, functions, and include files may use *MODULES* and *INTERFACES.* MODULES provide a mechanism for grouping subroutines, functions and data structures. Data and associated user-defined types may be declared exclusively for use within a module, with access to the "public" data defined by an INTERFACE. MODULEs and INTERFACEs also provide for overloading of both operators and procedures. For example, the "+" operator can be extended to add two variables of a user-defined type.

2.5 Dynamic memory allocation

The new Fortran standard supplies a restricted form of pointers for dynamic memory allocation. It is restricted in that a Fortran 90 *POINTER* may only point to a variable which is declared to be a *TARGET* or to dynamically allocated data. Any type of data, including arrays, may be allocated and assigned to a pointer using the *ALLOCATE* statement. Allocated memory is freed using the *DEALLOCATE* statement.

2.6 Computational precision

FORTRAN 77 contained two floating point data types with different numerical precision. Fortran 90 allows the user to specify the precision required for REAL, COMPLEX, INTEGER, and LOGICAL types by using the KIND parameter. This can, for example, be used to guarantee in a portable way that COMPLEX calculations are always done with 64-bit precision (assuming that an implementation provides that precision).

2.7 Syntax

In addition to the traditional FORTRAN fixed form (*i.e.* statement numbers in first 5 columns, code beginning in column 6, *etc.*), Fortran 90 provides a free source form. There are no column-oriented restrictions in the free form. In this new form, continuations are marked with a "&" at the end of a line.

3 Surveys

3.1 Vendor Survey

Two types of vendors were surveyed. Hardware vendors (Sect. 3.1.1) were queried as to their plans to provide full or partial Fortran 90 implementations with their machines. Third-party software vendors (Sect. 3.1.2) were asked about their implementations of Fortran 90, and the availability for various platforms.

3.1.1 Hardware vendors

The hardware vendors surveyed included manufacturers of HSPs (High Speed Processors), MPPs (Massively Parallel Processors), and workstations. These vendors were asked whether they planned to implement Fortran 90, and if so, whether they planned a full or partial implementation of the standard. If a partial implementation was planned, we attempted to determine which parts of the standard would be provided. We also requested estimated dates for completion of all or part of the compiler.

We were also interested in what the vendors considered the important performance issues in implementing new language features. Such issues include the parallelization and vectorization of array constructs, and the vectorization of elements in arrays of user-defined data structures.

3.1.1.1 Convex

Convex will continue to add Fortran 90 features to their Fortran compiler in response to customer demand. This compiler runs on the existing vector mini-supercomputers, and will be available on the new MPPs. They already support the full array syntax, many of the new Fortran 90 intrinsic functions (such as the shift functions), and several new control structures.

3.1.1.2 Cray Research, Inc.

Cray Research, Inc. (CRI) currently supports much of the array syntax as part of its *cf77* Fortran compiling system. A number of Fortran 90 syntax extensions, such as END DO are also supported. They are currently working on a full implementation of Fortran 90 for the Y-MP and later machines (including their planned MPP). This new product will not be available on the Cray 2 or the X-MP. The tentative release date for this product is in the third quarter of Calendar Year (CY) 1993. The new product will be a single compiler, unlike *cf77*, which consists of a compiler and two pre-processors.

In addition to standard Fortran 90, the compiler will accept almost all current CRI FORTRAN extensions, excepting some minor features such as interpreting "DOUBLE" to be synonymous with "DOUBLE PRECISION."

3.1.1.3 Digital Equipment Corp.

Digital Equipment Corp. (DEC) is planning to implement a full Fortran 90 compiler. This compiler will also support VAX Fortran. No details were available on the release schedule for this product.

3.1.1.4 Hewlett-Packard Co.

Hewlett-Packard (HP) begins a phased implementation of the complete standard concurrent with the release of HP-UX 9.0. The compiler is tuned for maximum performance on the HP Apollo 9000 Series 700 and HP 9000 Series 800 product lines. This initial phase supports array syntax, including automatic arrays, allocatable arrays, and an un-nested block WHERE. They also support user-defined types and the new control constructs (*e.g.* CASE and DO WHILE). A number of CRI and VAX Fortran extensions are also provided. Additional Fortran 90 features will be added based on requests and needs of users.

3.1.1.5 Intel Supercomputer Systems Division

Intel is planning a phased development of a new HPF compiler. As noted above, HPF contains the whole of Fortran 90. The first phase, which will enter internal beta-testing in the fourth quarter of CY 1992, and be supported in the second quarter of CY 1993, will implement the HPF Fortran 90 subset. This will include the array syntax, the single-statement FORALL (a parallel construct which was considered for Fortran 90), and the WHERE statement. Intrinsic functions will accept array arguments, and the array syntax will result in some degree of automatic parallelization. HPF data distribution directives will also be included in this first release.

3.1.1.6 Kendall Square

Kendall Square plans to provide a full Fortran 90 implementation as a future deliverable. The current partial implementation includes the array syntax, WHERE statement, ALLOCATE and FREE statements, and new control syntax. They also provide many VAX Fortran extensions. Additional Fortran 90 features will be provided in a phased implementation of the standard.

3.1.1.7 MasPar

The MasPar Fortran compiler, based on a compiler originally supplied by Compass, Inc., partially supports Fortran 90. They plan to completely implement HPF, and thus, Fortran 90. The compiler currently supports the complete array syntax exclusive of allocatable arrays. Most of the array intrinsic functions have been added to the compiler, and the other intrinsic functions accept array arguments. MasPar does not currently support user-defined types, modules, or operator-overloading. MasPar supports the FORALL statement, and is working on improving the parallel optimization. MasPar places the highest priority on improving the performance of its compiler and implementing HPF features.

3.1.1.8 Silicon Graphics Inc.

Silicon Graphics Inc. (SGI) is planning a full implementation of Fortran 90. It is tentatively scheduled for completion in the summer of CY 1993, but may be finished later.

3.1.1.9 TERA Computer Co.

TERA Computer Corp. is building a parallel machine which consists of scalar processors, with a planned release date in 1993-94. They plan to implement a subset of Fortran 90 for their initial Fortran compiler. They plan to supply a subset of the array extensions; in particular, they are implementing array syntax and array section notation in expressions. The array expressions will automatically parallelize. The WHERE construct will be provided. They will supply commonly available extensions such as the DO WHILE statement.

3.1.1.10 Thinking Machines Corp.

Thinking Machines Corp. (TMC) supports a compiler (CM Fortran, or CMF) which, like the MasPar compiler, is derived from the Compass, Inc. compiler. CMF currently provides a subset of Fortran 90. With release 2.0 of CMF, the array syntax is completely implemented, including nested WHERE expressions and allocatable arrays. CMF intrinsic functions accept array arguments. Namelist I/O has also been implemented. POINTERS without multiple assignments may be available later this year. Implementation of STRUCTUREs and MODULEs has not yet been scheduled. TMC supports the FORALL statement. They are planning to implement HPF, which will entail incorporating the HPF Fortran 90 subset in a timely fashion, and ultimately providing a complete Fortran 90 implementation. Local functions, a TMC proposal to HPF, will be implemented within the next year. Sixty-four bit integers will also be supported.

3.1.2 Third party vendors

Five third-party vendors were contacted for this survey. One vendor supplies compilers principally for Multiple Instruction stream Multiple Data stream (MIMD) MPPs. Another provides a parallel-code generation tool for MIMD message passing and Fortran 90 array syntax. The remaining three vendors are selling complete, or nearly complete, implementations of Fortran 90. One of these (NAGWare f90) uses C as the intermediate language, while the other two (Parasoft and Pacific-Sierra) use FORTRAN 77. Intermediate source files may be saved for all three products. While the intermediate language is portable, the products rely on libraries which are platform-dependent. All products supply *man* pages.

3.1.2.1 Applied Parallel Research

Applied Parallel Research is developing a new version of *FORGE*, a sequential-to-parallel program conversion tool for distributed and shared memory machines. *FORGE 90* will produce code for Ardent, Convex, CRI, Intel, nCUBE, and TMC machines, among others. In addition to Express and PVM message passing code may also be produced. A CMF code generator is available, with a limited implementation of HPF planned for November. Full support for Fortran 90 and HPF is expected in December 1993.

3.1.2.2 NAGWare f90

NAGWare f90 is a complete, standard-conforming implementation of Fortran 90. It uses Kernighan and Ritchie [4] C as an intermediate language, and is available on Apollo, DEC (DECStation, Ultrix VAX, & VMS VAX), HP, IBM, NeXT, SGI, and Sun workstations. It is also available for 386 and 486 personal computers. Several large codes have been ported to Fortran 90 using this product on an Apollo workstation [5]. NAG sells a Fortran 90 test suite.

3.1.2.3 Pacific-Sierra VAST-90

Pacific-Sierra implements the entire Fortran 90 standard, except for non-advancing I/O which cannot be provided in this type of translator. Additionally, VAST-90 will translate FORTRAN 77 code into Fortran 90. This consists, in part, of substituting array constructs for DO loops (when possible), replacing COMMONs with MODULEs, and replacing GOTOs with CYCLEs, EXITs, and BLOCK IFs. The product is available on Sun, DEC (Ultrix and VAX), HP and IBM workstations. It is also available on Convex computers. They would be interested in porting the product to CRI machines.

3.1.2.4 ParaSoft f90

Parasoft f90 is a complete, standard-conforming implementation of Fortran 90, and also supports CM Fortran. Various vendor extensions to FORTRAN 77 not contained within Fortran 90 are also supported. The user may instruct f90 to flag non-standard extensions. The product uses FORTRAN 77 as an intermediate language. This product is available on Sun, SGI, HP, DEC, and IBM workstations. It is also available on Convex mini-supercomputers, the IBM 3090 mainframe, and the Cray 2, X-MP, and Y-MP.

3.1.2.5 The Portland Group

The Portland Group is in the process of implementing HPF. The first version of this compiler is expected in the middle of CY 1993. This release will support the HPF Fortran 90 subset, in addition to HPF data distribution directives. Full HPF/Fortran 90 compliance may occur by the end of 1994. The target market for this compiler is MIMD MPPs. The HPF compiler is designed to produce portable FORTRAN 77 combined with message passing; it will also produce assembly language under certain architectures.

This compiler should be available on the Intel iPSC860 and Paragon machines. It will be targeted to other specific architectures, such as SPARC- and Fujitsu VPU-based MIMD systems and workstations.

3.2 User Survey

Nine users and managers were contacted for this survey. The most senior contact was a code R branch chief, the most junior was a Computational Fluid Dynamics (CFD) researcher working as a contractor.

Two contacts, both from RFTT, were not interested in using Fortran 90 on the High Speed Processors (HSPs). They presently use Vectoral rather than Fortran. They believe that the improvements pertaining to structured programming are already present in Vectoral. Furthermore, variable-length and size arrays are not present in Fortran 90, although they are present in Vectoral. The lack of nested procedures was also cited as a reason not to use Fortran 90.

Portability was an important consideration for the surveyed users, with approximately equal interest in seeing Fortran 90 implemented on HSPs, massively parallel processors (MPPs), and workstations. Three contacts (two from RFTC and one from RAA) specifically required compatibility with CM Fortran.

Four contacts (from RFA, RFTT, and RAA) explicitly stated that they preferred adoption of the entire standard. They considered the possibility of varying vendor-selected Fortran 90 subsets to be an obstacle to portability.

Two contacts (from RN and RFA) expected to write new code in either C or C++. This code is related to graphics applications. These users would not consider using Fortran 90 for this code if it were available, although they both were interested in using Fortran 90 features to maintain and improve existing FORTRAN 77 codes.

Three contacts (from RFA, RAA, and RFTC) stated that Fortran 90 was a much better path than C or C++ for introducing new language features into scientific application codes at NAS. This was due to the ability to incrementally adopt new features, while maintaining compatibility with FORTRAN 77.

Unsurprisingly, most contacts were not familiar with all of the new features of Fortran 90. When asked to select new features which appealed to them, array constructs and user-defined types were mentioned twice, and the new control constructs were mentioned once. One user, who was interested in the array syntax, stated that a poor (specifically, a poorly vectorized) implementation of the array constructs would discourage him from using that part of the standard in his code. Three users liked the new features in general, without picking out any one in particular.

4 Conclusions and Recommendations

Computer hardware vendors are clearly moving toward the adoption of Fortran 90. Partial implementations are available today, and several fully compliant compilers will be in use by the end of CY 1993. Moreover, three third party compilers are on the market today for a variety of machines, including all NAS computers except for the MPPs.

Users, with the exception of Vectoral users, appear to be interested in using Fortran 90 on the full range of NAS machines. While no one we contacted was planning to adopt all of the new features simultaneously, there was strong interest in having the complete standard language available. While some users are adopting C or C++, there was more interest in the gradual changes represented by Fortran 90.

Most vendors surveyed plan to implement the whole of Fortran 90. However, the completion date for most of these vendors is rather uncertain. While partial implementations are useful, users will not be guaranteed portability until they can rely on using the complete standard on all relevant hardware platforms. The early completion of the HPF Fortran 90 subset on the MPP vendors' machines will ameliorate this situation, but true portability relies on the complete standard.

In order to ensure that Fortran 90 is available on machines ranging from workstations to HSPs and MPPs, NAS should take the measures given below. In particular, complete Fortran 90 compliance by late 1994 or 1995 should be specified in the procurement process for new machines. This will encourage vendors with incomplete plans for Fortran 90 to implement the entire standard. It will also encourage those already planning complete implementations to accomplish those plans in a timely manner. If possible, compiler upgrades or replacements should be pursued for those machines currently installed at NAS which are likely to remain in use in 1995.

As an interim measure, NAS should consider acquiring third party Fortran 90 compilers. As described above, several products are available for a variety of platforms. All of these us an intermediate high-level language in their compilation process. This may degrade performance relative to native C or FORTRAN compilers. It may be prudent to test all available compilers for each hardware platform in order to provide users the best possible performance.

In order to assure compliance with Fortran 90, a standard test suite should be acquired. One such test suite is available from NAG. NAS should determine what similar suites, if any, are available elsewhere, and acquire the best available product.

Acknowledgements

I would like to acknowledge the assistance which I received from vendors and NAS users in preparing this report. David Browning of NAS RND provided invaluable assistance in evaluating new language features.

References

[1] International Standard Programming Language Fortran, ISO/IEC 1539: 1991 (E). ISO/IEC Copyright Office, Case Postale 56, CH-1211 Geneve 20, Switzerland.

American National Standard Programming Language FORTRAN, ANSI X3.198 - 1992. American National Standards Institute, 1430 Broadway, New York, NY 10018.

[2] American National Standard Programming Language FORTRAN, ANSI X3.9 - 1978. American National Standards Institute, 1430 Broadway, New York, NY 10018.

[3] W. Brainard, C. Goldberg, and J. Adams, *A Programmer's Guide to Fortran 90*, McGraw-Hill Book Co., New York, N.Y. 1991.

Michael Metcalf and John Reid, *Fortran 90 Explained*, Oxford University Press, Oxford, U.K., 1991.

[4] Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language*, Prentice-Hall, Inc., Englewood Cliffs, N.J. 07632. 1978.

[5] Michael Metcalf, "A First Encounter with f90," **Fortran Forum 11** (1992) 24.